

Chapter 9

Arrays

GOALS:

1. Students will be able to create and use Arrays.
2. Students will be able to create and use Two-Dimensional arrays.

ESSENTIAL QUESTIONS

1. What are some advantages of using arrays?
2. Why is it necessary to use arrays?
3. How can you use 2D Arrays to represents board games?

LESSON

A Variable Array stores a set of variables that each have the same name and are all of the same type.

Member/Element – variable of an array.

Index – number that identifies each member of an array usually starting at 0 unless specified otherwise.

Option Base – Placed in the General Section and is used to change the index from 0 to 1.

Example of an Array:

0	1	2	3	4
Bob	Sam	Sue	Kim	Abe

Array Declaration

An array is declared with a Dim, Private, or Static statement that includes the identifier followed by the upper bound of the array in parentheses and the data type. Example: Dim strName(5) as String ‘ 5 elements

Lower Bound – When declaring arrays, you can specify the start index by using the lower bound to declare an array. Example: Dim strNames (1 to 5) as String ‘Where 1 is the lower bound instead of 0

Array initialization

When an array is declared, the element variables are set to default values. Numeric variables are set to 0. Strings are set to empty Strings and Boolean array elements to False values.

Accessing an element

To gain access to an element of an array, you must use the parentheses. Example: lblAns.Caption = strName(4) Display the name in the 5th position since it starts at zero.

Assigning a value to an array

You must use the array name with the index number in the parentheses. Example: strname(3) = “Bobby”

LBound and UBound

The LBound and UBound are built-in functions used to determine the lower and upper bound of an array. Examples: LBound(ArrayName) or UBound(ArrayName)

For...Next Loop

The For...Next loop is often used for arrays because of the counter of the loop can be used to access the indexes. Example:

```
For intIndex = LBound(ArrayName) To UBound(ArrayName)
    ArrayName(intIndex) = InputBox("Enter name", "Student")
Next intIndex
```

Range Errors

A run-time error occurs if code refers to an index outside the lower bound to upper bound range of an array. The error message produced is "Subscript out of range". Subscript is the same as index.

Project 1

1. Create the follow program
 2. Answer questions at the end of project.
- ' Display Names

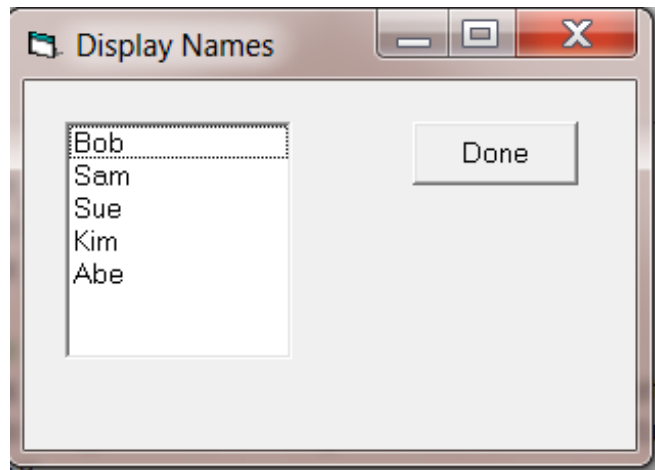
Option Explicit

```
Private Sub Form_Load()
    Dim strNames(1 To 5) As String
    Dim lngIndex As Long

    ' Load names
    For lngIndex = LBound(strNames) To
        UBound(strNames)
        strNames(lngIndex) = InputBox("Enter student's first name:", "Students")
    Next lngIndex

    ' Display names
    For lngIndex = LBound(strNames) To UBound(strNames)
        lstNames.AddItem strNames(lngIndex)
    Next lngIndex
End Sub

Private Sub cmdDone_Click()
    Unload Me
End Sub
```



Questions

1. Why did the program use two For Loops?
2. What is the purpose of LBound() & UBound() procedures?
3. What is the purpose of lngIndex variable?
4. How many elements were created with the statement: Dim strNames(1 To 5) As String?

ARRAY PARAMETERS

Arrays can be passed as arguments and declared as procedure parameters by using the array name followed by an empty set of parentheses. However, arrays are required to be passed by reference (ByRef).

Example:

```
Function SumOfItems(ByRef intNumArray() As Integer) As Long
    Dim lngIndex As Long
    Dim lngSum As Long
    lngSum = 0
    For lngIndex = LBound(intNumArray) To UBound(intNumArray)
        lngSum = lngSum + intNumArray(lngIndex)
    Next lngIndex
    SumOfItems = lngSum
End Function
```

Notice the Function passes the array intNumArray() by reference.

The array name followed by an empty set of parentheses is used in a procedure call.

Example:

```
lngTotal = SumOfItems(intDataArray())
```

The array intDataArray() is passed through the Function Parameter list.

Single Element parameter

A single element of an array can be passed by value or reference as an argument.

Example:

```
Sub DisplayElement(ByVal intNumber As Integer, ByRef lblLabel as Label)
    lblLabel.Caption = intNumber
End Sub
```

Call DisplayElement(intDataArray(3)) ' The 3rd element of the array is passed

Arrays with Meaningful Indexes

It is always a good idea to use meaningful names for array indexes.

Project 2

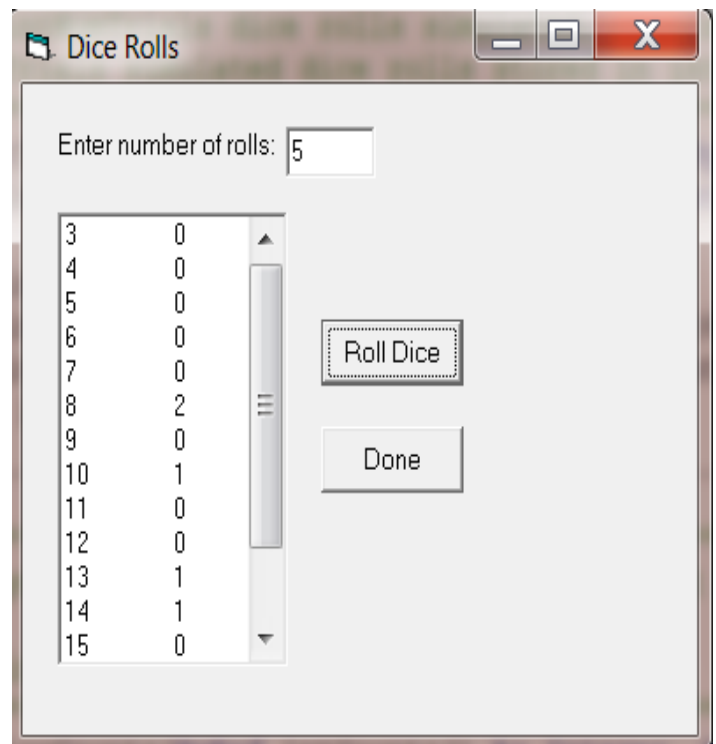
1. Create the following program
 2. Answer the Questions at the end
- ' Dice Rolls

Option Explicit

```
Private Sub Form_Load()
    Randomize
End Sub
```

```
Private Sub txtNumRolls_Change()
    lstOutcome.Clear
End Sub
```

```
Private Sub cmdRollDice_Click()
    Dim intTrials As Integer
    Dim intCounts(3 To 18) As Integer
```



```

intTrials = txtNumRolls.Text
Call CountTrials(intTrials, intCounts())
Call DisplayCounts(intCounts(), lstOutcome)
End Sub

```

```

'*****

```

```

' Simulates intNumTrials rolls of three dice and stores the
' counts of each outcome in intCounts()
',

```

```

' pre: Randomize has been called
' post: intNumTrials dice rolls simulated. Counts of
' intNumTrials simulated dice rolls stored in intCounts()
'*****

```

```

Sub CountTrials(ByVal intNumTrials As Integer, ByRef intCounts() As Integer)
    Dim intRoll As Integer, intRollOutcome As Integer

```

```

    For intRoll = 1 To intNumTrials
        intRollOutcome = Int(6 * Rnd + 1) + Int(6 * Rnd + 1) + Int(6 * Rnd + 1)
        intCounts(intRollOutcome) = intCounts(intRollOutcome) + 1
    Next intRoll
End Sub

```

```

'*****

```

```

' Displays the contents of intCounts() in a list box
',

```

```

' post: intCounts() displayed in a list box
'*****

```

```

Sub DisplayCounts(ByRef intCounts() As Integer, ByRef lstList As ListBox)
    Dim intRollOutcome As Integer

```

```

    For intRollOutcome = LBound(intCounts) To UBound(intCounts)
        lstList.AddItem intRollOutcome & vbTab & intCounts(intRollOutcome)
    Next intRollOutcome
End Sub

```

```

Private Sub cmdDone_Click()
    Unload Me
End Sub

```

Questions:

1. What is the purpose of the program?
2. How was the variable intTrials passed to the CountTrials() procedure?
3. How was the array intCount() passed to the CountTrials() procedure?

Project 3

1. Create the following program.
2. Answer the Questions at the end of project

' Number Occurrences

Option Explicit

```
Private Sub Form_Load()
```

```
    Randomize
```

```
End Sub
```

```
Private Sub cmdCountNumerals_Click()
```

```
    Dim strNumber As String
```

```
    Dim intNumberCounts(48 To 57) As Integer ' 48 is ASCII code for 0,  
                                                ' 57 is ASCII code for 9
```

```
    strNumber = txtNumberString.Text
```

```
    Call CountNumbers(strNumber, intNumberCounts())
```

```
    Call DisplayCounts(intNumberCounts(), lstOutcome)
```

```
End Sub
```

Digit	Count
0	0
1	4
2	4
3	5
4	0
5	3
6	1
7	1
8	0
9	0

```
*****
```

```
' Counts the occurrences of numbers 0 through 9, in strNumber
```

```
,
```

```
' post: intNumberCounts() contains the counts of the
```

```
' occurrences of numbers 0 through 9 in strNumber
```

```
*****
```

```
Sub CountNumbers(ByVal strNumber As String, ByRef intNumberCounts() As Integer)
```

```
    Dim intNumber As Integer
```

```
    Dim strNumberCharacter As String
```

```
    For intNumber = 1 To Len(strNumber)
```

```
        strNumberCharacter = Mid(strNumber, intNumber, 1)
```

```
        If strNumberCharacter >= "0" And strNumberCharacter <= "9" Then
```

```
            intNumberCounts(Asc(strNumberCharacter)) = intNumberCounts(Asc(strNumberCharacter)) + 1
```

```
        End If
```

```
    Next intNumber
```

```
End Sub
```

```
*****
```

```
' Displays the contents of intNumberCounts() in a list box
```

```
,
```

```
' post: intNumberCounts() displayed in a list box
```

```
*****
```

```
Sub DisplayCounts(ByRef intNumberCounts() As Integer, ByRef lstList As ListBox)
```

```
    Dim intNumberCount As Integer
```

```
    lstList.Clear
```

```
    For intNumberCount = LBound(intNumberCounts) To UBound(intNumberCounts)
```

```
        lstList.AddItem Chr(intNumberCount) & vbTab & intNumberCounts(intNumberCount)
```

```
    Next intNumberCount
```

```
End Sub
```

```
Private Sub cmdDone_Click()
```

```
    Unload Me
```

```
End Sub
```

Questions

1. Why did the array start at 48 to 57 instead of starting at 0 or 1? “intNumberCounts(48 To 57)”

2. What is the purpose of the following statement?

```
intNumberCounts(Asc(strNumberCharacter)) = intNumberCounts(Asc(strNumberCharacter)) + 1
```

3. What is the purpose of the following statement?

```
strNumberCharacter = Mid(strNumber, intNumber, 1)
```

Searching an Array

Linear Search – Works by proceeding from one element to the next through the array until the desired value is found or the entire array has been searched.

The Function below searches an array

```
Function FindItemIndex(ByRef intDataArray() as Integer, ByVal intSearchItem as Integer) as Integer
```

```
    Dim intIndex as Integer
```

```
    intIndex = LBound(intDataArray)
```

```
    Do While (intDataArray(intIndex) <> intSearchItem) And (intIndex < UBound(intDataArray))
```

```
        intIndex = intIndex + 1
```

```
    Loop
```

```
    If intDataArray(intIndex) = intSearchItem Then
```

```
        FindItemIndex = intIndex          ‘Item found
```

```
    Else
```

```
        FindItemIndex = -1                ‘Item not found
```

```
    End if
```

```
End Function
```

Dynamic Arrays

A Dynamic Array – can vary in size during run time and is used when the size of an array is unknown at the start of the program or when the size of the array changes throughout the run of the program.

Declaration Example:

```
Dim intDataArray() as Integer
```

Notice there is no value between the parentheses meaning the size is unknown.

ReDim – statement that allocates space for the elements of a dynamic array.

The upper bound for the array must be included. The lower bound may also be included.

Example:

```
ReDim intDataArray(1 To 5)
```

You can use ReDim over and over again to resize the array. However, each time you ReDim an array all data is lost.

ReDim Preserve – allows you to keep old data when resizing only changing the upper bound. You get an error if you try to change the lower bound. You cannot use ReDim Preserve until the dynamic array been dimensioned (ReDim).

Project 4

1. Create the following program
2. Answer all questions at the end of project.

' Find Name

Option Explicit

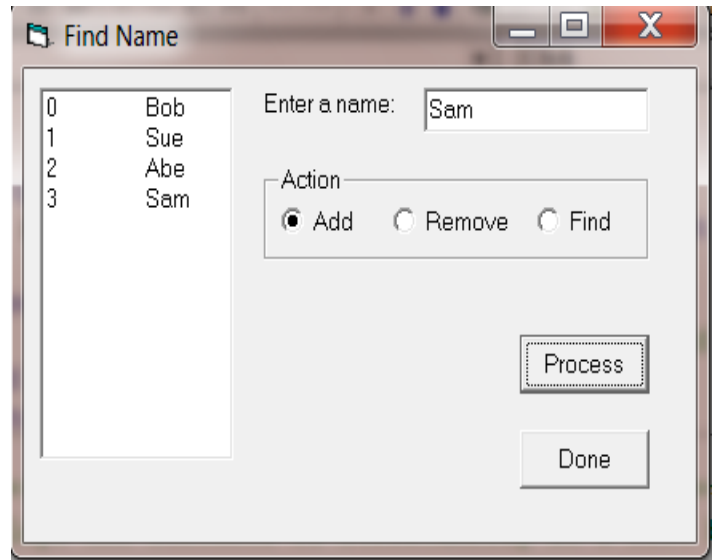
```
Private Sub Form_Load()
    Randomize
    optAdd.Value = True
    optRemove.Value = False
    optFind.Value = False
End Sub
```

```
Private Sub cmdProcess_Click()
    Static strDataArray() As String
    Static intNumDataItems As Integer
    Dim strNameEntered As String
    Dim intIndex As Integer
```

```
    strNameEntered = txtNameEntered.Text
    If optAdd.Value Then
        Call AddItem(strDataArray(), intNumDataItems, strNameEntered)
        Call DisplayData(strDataArray(), intNumDataItems, lstArrayElements)
    ElseIf optRemove.Value And intNumDataItems > 0 Then
        Call RemoveItem(strDataArray(), intNumDataItems, strNameEntered)
        Call DisplayData(strDataArray(), intNumDataItems, lstArrayElements)
    ElseIf optFind.Value And intNumDataItems > 0 Then
        intIndex = FindItemIndex(strDataArray(), strNameEntered)
        If intIndex = -1 Then
            lstArrayElements.AddItem "Item not found"
        Else
            lstArrayElements.AddItem "Item at index " & intIndex
        End If
    End If
End Sub
```

```
*****
' Increases size of strDataArray by 1, increments intNumDataItems
' by 1, and adds strNameToAdd as the last element of the array
'
' pre: Lower bound is 0
' post: intDataArray increased by 1, intNumDataItems incremented
' by 1, and strNameToAdd last element of array
*****
```

```
Sub AddItem(ByRef strDataArray() As String, _
    ByRef intNumDataItems As Integer, ByVal strNameToAdd As String)
    If intNumDataItems = 0 Then
```



' Number of items in strDataArray

```

        ReDim strDataArray(intNumDataItems)
    Else
        ReDim Preserve strDataArray(intNumDataItems)
    End If
    strDataArray(intNumDataItems) = strNameToAdd
    intNumDataItems = intNumDataItems + 1
End Sub

'*****
' Decreases size of strDataArray by 1 and decrements
' intNumDataItems by 1
',
' pre: intNumDataItems > 0
' post: strDataArray decreased by 1, intNumDataItems
' decremented by 1
'*****

Sub RemoveItem(ByRef strDataArray() As String, _
ByRef intNumDataItems As Integer, ByVal strNameToRemove As String)
    Dim intItemIndex As Integer, intIndex As Integer

    intItemIndex = FindItemIndex(strDataArray(), strNameToRemove)
    If intItemIndex > -1 Then ' fill spot of removed item
        For intIndex = intItemIndex To (UBound(strDataArray) - 1)
            strDataArray(intIndex) = strDataArray(intIndex + 1)
        Next intIndex
        intNumDataItems = intNumDataItems - 1 ' decrement strNumDataItems
        If intNumDataItems > 0 Then ' resize array
            ReDim Preserve strDataArray(intNumDataItems - 1)
        End If
    End If
End Sub

'*****
' Returns the index of the first occurrence of strSearchItem in
' strDataArray or -1 if strSearchItem not found
',
' pre: intDataArray has at least one element
' post: Index of the first occurrence of strSearchItem returned, or
' -1 returned if strSearchItem not found.
'*****

Function FindItemIndex(ByRef strDataArray() As String, _
ByVal strSearchItem As String) As Integer
    Dim intIndex As Integer

    intIndex = LBound(strDataArray)
    Do While (strDataArray(intIndex) <> strSearchItem) _
        And (intIndex < UBound(strDataArray))
        intIndex = intIndex + 1
    Loop
    If strDataArray(intIndex) = strSearchItem Then
        FindItemIndex = intIndex ' Item found
    Else

```

```

        FindItemIndex = -1
    End If
End Function

' Item not found

' Displays the contents of strDataArray in a list box
'
' post: strDataArray() displayed in a list box
Sub DisplayData(ByRef strDataArray() As String, _
    ByVal intNumDataItems As Integer, ByVal lstList As ListBox)
    Dim intIndex As Integer

    lstList.Clear
    If intNumDataItems > 0 Then
        For intIndex = LBound(strDataArray) To UBound(strDataArray)
            lstList.AddItem intIndex & vbTab & strDataArray(intIndex)
        Next intIndex
    End If
End Sub

Private Sub cmdDone_Click()
    Unload Me
End Sub

```

Questions

1. What Procedure or Function runs the entire program?
2. Why was ReDim used in the AddItem and RemoveItem sub procedures?

TWO-DIMENSIONAL

A matrix is a multidimensional variable array that is used to store related information. A 2D array can be used to represent data that relates to a grid.

Declaring a 2D Array:

A matrix is declared with a Dim, Private, or Static statement that includes the identifier following by the upper bound of the first dimensional (the rows) and the upper bound of the second dimensional (the columns) separated by a comma in parentheses and then the data type.

Example:

```
Dim strTTTBoard(2,2) As String
```

Declares a 3x3 matrix.

	0	1	2
0			
1			
2			

Assigning Elements of a 2D Array

Each element can be assigned a value using the equal symbol (=) with the name of 2D array and row and column number in the parentheses. Example:

```
strTTTBoard(0,1) = cmdBoard.Caption
```

The lower bound of each dimension of a matrix can be specified in the matrix declaration.

Example:

```
Dim strTTTBoard(1 To 3, 1 To 3) As String
```

LBound & UBound

The LBound and UBound VB Built-in functions may also be used with matrices to determine lower and upper bounds of the dimensions.

Examples:

LBound(MatrixName, Dimension) 'Dimension is 1 for row dimension or 2 for column dimension

UBound(MatrixName, Dimension)

Nested Loops

Nested For...Next loops are often used to access the elements of a matrix because the loop counter indicates the row of the matrix and the other counter indicates the columns.

Example:

```
Dim strTTTBoard(2,2) as String
```

```
For lngRow = LBound(strTTTBoard, 1) To UBound(strTTTBoard, 1)
```

```
    For lngCol = LBound(strTTTBoard, 2) To UBound(strTTTBoard, 2)
```

```
        lstMoveAddItem strTTTBoard(lngRow, lngCol)
```

```
    Next lngCol
```

```
Next lngRow
```

Dynamic Matrix

A matrix may also be dynamic. As with a one-dimensional dynamic array, a dynamic matrix is declared in a Dim statement with an empty set of parentheses and then a ReDim statement is later used to change the dimensions.

1. Create the following Program'
2. Answer the following questions

Option Explicit

```
*****
```

```
' Two players play a game of Tic-Tac-Toe
```

```
,
```

```
' pre: Board is made of a command button control array with
```

```
' empty Captions and Indexes in the following order:
```

```
'    0 1 2
```

```
'    3 4 5
```

```
'    6 7 8
```

```
' post: Tic-Tac-Toe has been played until a winner or a
```

```
' draw is declared or the Done button clicked.
```

```
*****
```

```
Private Sub cmdTTTSquares_Click(Index As Integer)
```

```
    Static strTTT(2, 2) As String
```

```
    ' Store players moves
```

```
    Static strPlayer As String
```

```
    ' O or X
```

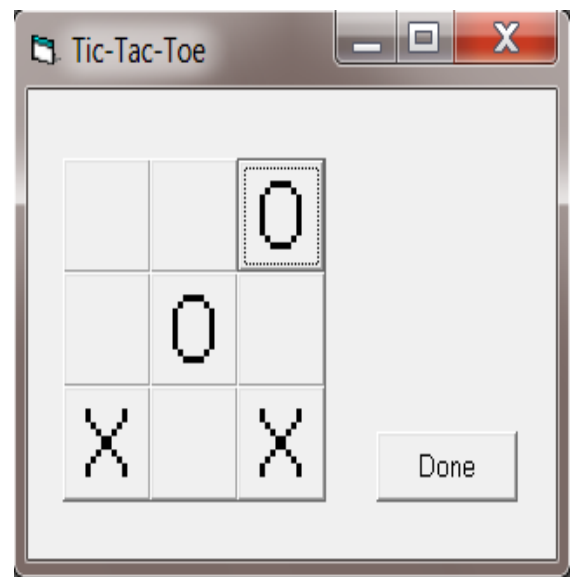
```
    ' Initialize strPlayer
```

```
    If strPlayer = "" Then
```

```
        strPlayer = "X"
```

```
    ' X goes first
```

```
End If
```



```

' Check for existing X or O
If cmdTTTSquares(Index).Caption <> "" Then
    MsgBox "Invalid move."
Else
    ' Show move
    cmdTTTSquares(Index).Caption = strPlayer

    ' Store move in strTTT()
    Call MakeMove(Index, strPlayer, strTTT())

    ' Check for winner
    If IsWinner(strTTT()) Then
        MsgBox "Game over!"
        Call NewGame(strPlayer, strTTT(), cmdTTTSquares)
    Else
        ' Next player's turn
        If strPlayer = "X" Then
            strPlayer = "O"
        Else
            strPlayer = "X"
        End If
    End If
End If
End Sub

'*****
' Store Tic-Tac-Toe move in strTTT() matrix
'
' pre: 0 <= intIndex <= 8, strTTT() is 3 x 3 matrix with
' indexing starting at 0
' post: Tic-Tac-Toe move stored in strTTT() matrix
'*****
Sub MakeMove(ByVal intIndex As Integer, ByVal strPlayer As String, _
ByRef strTTT() As String)
    If intIndex <= 2 Then
        strTTT(0, intIndex) = strPlayer
        ' Move made in first row
    ElseIf intIndex <= 5 Then
        strTTT(1, intIndex - 3) = strPlayer
        ' Move made in second row
    Else
        strTTT(2, intIndex - 6) = strPlayer
        ' Move made in third row
    End If
End Sub

'*****
' Determines if there is a winner
'
' pre: strTTT() is 3 x 3 matrix with indexing starting at 0
' post: True returned if a winner is found or if all the
' TTTSquares are filled
'*****
Function IsWinner(ByRef strTTT() As String) As Boolean
    Dim intRow As Integer, intCol As Integer
    Dim blnMovesLeft As Boolean

```

```

IsWinner = False

For intRow = 0 To 2                                     ' Check all rows
    If strTTT(intRow, 0) = strTTT(intRow, 1) And _
        strTTT(intRow, 1) = strTTT(intRow, 2) And _
        strTTT(intRow, 0) <> "" Then
        IsWinner = True
    End If
Next intRow

For intCol = 0 To 2                                     ' Check all columns
    If strTTT(0, intCol) = strTTT(1, intCol) And _
        strTTT(1, intCol) = strTTT(2, intCol) And _
        strTTT(0, intCol) <> "" Then
        IsWinner = True
    End If
Next intCol

' Check one diagonal
If strTTT(0, 0) = strTTT(1, 1) And strTTT(1, 1) = strTTT(2, 2) And _
    strTTT(0, 0) <> "" Then
    IsWinner = True
End If

' Check other diagonal
If strTTT(0, 2) = strTTT(1, 1) And strTTT(1, 1) = strTTT(2, 0) And _
    strTTT(0, 2) <> "" Then
    IsWinner = True
End If

' Check for empty squares
blnMovesLeft = False
For intRow = 0 To 2
    For intCol = 0 To 2
        If strTTT(intRow, intCol) = "" Then
            blnMovesLeft = True
        End If
    Next intCol
Next intRow
If Not blnMovesLeft Then
    IsWinner = True
End If
End Function

*****

' Changes player to X, reinitializes strTTT(), and
' Clears cmdTTTSquares
,

' pre: strTTT() is 3 x 3 matrix with indexing starting at 0 and
' cmdSquares is a command button control array of 9 buttons with
' indexing starting at 0

```

```

' post: strTTT() matrix initialized and control array captions cleared
*****

Sub NewGame(ByRef strPlayer As String, ByRef strTTT() As String, _
ByRef cmdTTTSquares As Object)
    Dim intIndex As Integer, intRow As Integer, intCol As Integer

    ' Player X starts
    strPlayer = "X"

    ' Clear player moves
    For intRow = 0 To 2
        For intCol = 0 To 2
            strTTT(intRow, intCol) = ""
        Next intCol
    Next intRow

    ' Clear board
    For intIndex = 0 To 8
        cmdTTTSquares(intIndex).Caption = ""
    Next intIndex
End Sub

Private Sub cmdDone_Click()
    Unload Me
End Sub

Private Sub Form_Load()

End Sub

```

Questions:

1. Why must you use the nested For...Loop when working with 2D Arrays?

